Analysis and design of software ecosystem architectures – towards the 4S telemedicine ecosystem

Christensen, Henrik Bærbak; Hansen, Klaus Marius; Kyng, Morten; Manikas, Konstantinos

Published in:
Information and Software Technology

DOI:
10.1016/j.infsof.2014.05.002

Publication date:
2014

Document version
Publisher's PDF, also known as Version of record

Information and Software Technology 56 (2014) 1476–1492

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Analysis and design of software ecosystem architectures – Towards the 4S telemedicine ecosystem

Henrik Bærbak Christensen [a], Klaus Marius Hansen [b,*], Morten Kyng [a,c], Konstantinos Manikas [b]

[a] Department of Computer Science, Aarhus University, Denmark
[b] Department of Computer Science (DIKU), University of Copenhagen, Denmark
[c] The Alexandra Institute, Aarhus, Denmark

ARTICLE INFO

Article history:
Received 4 July 2013
Received in revised form 8 May 2014
Accepted 9 May 2014
Available online 28 May 2014

Keywords:
Software ecosystem architecture
Third-party sponsored software ecosystems
Telemedicine software ecosystems

ABSTRACT

Context: Telemedicine, the provision of health care at a distance, is arguably an effective way of increasing access to, reducing cost of, and improving quality of care. However, the deployment of telemedicine is faced with standards that are hard to use, application-specific data models, and application stove-pipes that inhibit the adoption of telemedical solutions. To which extent can a software ecosystem approach to telemedicine alleviate this?
Objective: In this article, we define the concept of *software ecosystem architecture* as the structure(s) of a software ecosystem comprising elements, relations among them, and properties of both. Our objective is to show how this concept can be used (i) in the analysis of existing software ecosystems and (ii) in the design of new software ecosystems.
Method: We performed a mixed-method study that consisted of a case study and an experiment. For (i), we performed a descriptive, revelatory case study of the Danish telemedicine ecosystem and for (ii), we experimentally designed, implemented, and evaluated the architecture of 4S.
Results: We contribute in three areas. First, we define the software ecosystem architecture concept that captures organization, business, and software aspects of software ecosystems. Secondly, we apply this concept in our case study and demonstrate that it is a viable concept for software ecosystem analysis. Finally, based on our experiments, we discuss the practice of software engineering for software ecosystems drawn from experience in creating and evolving the 4S telemedicine ecosystem.
Conclusion: The concept of software ecosystem architecture can be used analytically and constructively in respectively the analysis and design of software ecosystems.

1. Introduction

The research field of *software ecosystems* has emerged as the study of the complex interaction between extensible software frameworks and software architecture(s) on one hand, and organizations, users, customers, developers, and businesses on the other. It is inspired by natural ecosystems in which species are characterized by symbiotic relationships and their survival relies heavily on the survival of the ecosystem [45,33,6,10,11,39]. We define a 'software ecosystem' as:

> the interaction of a set of actors on top of a common technological platform that results in a number of software solutions or services. [42].

Further, "Each actor is motivated by a set of interests or business models and connected to the rest of the actors and the ecosystem as a whole with symbiotic relationships, while, the technological platform is structured in a way that allows the involvement and contribution of the different actors." [42].

A well-known example of a software ecosystem is the Android ecosystem. From a software ecosystem point of view, Google controls the Android platform while external developers can build applications ("apps") that are distributed to Android users via the Google Play store. Thus, Google has collaborated with external developers to quickly build functionality in the form of more than 700,000 apps [64]. In this way, the Android software ecosystem has arguably helped Google increase the value of Android for its users, increased attractiveness, accelerated innovation, and decreased cost [6].

We may distinguish between two main elements of software ecosystems:

* Corresponding author. Tel.: +45 61732721.
E-mail addresses: hbc@cs.au.dk (H.B. Christensen), klausmh@diku.dk (K.M. Hansen), mkyng@cs.au.dk (M. Kyng), kmanikas@diku.dk (K. Manikas).

http://dx.doi.org/10.1016/j.infsof.2014.05.002
0950-5849/

- *Actors* that may, e.g., be individuals or organizations. Depending on their activities in the ecosystem, they can have different roles including 'orchestrator' (such as Google in the Android ecosystem), 'keystone' (such as Samsung in the Google ecosystem), and 'niche player' (such as external software developers in the Android ecosystem). Additionally, each actor has an incentive for being active in the ecosystem that, often, can be represented by a business model.
- *Software* that exists as a platform/framework or as software solutions or services built on the platform. Software forms a main element of software ecosystems and a software ecosystem can be studied through the software elements that it consists of, their relations, and properties. The ability of the platform to incorporate different elements and the interaction and interoperability of the elements, are characteristics of the platform.

In this article we report on work in relation to telemedicine ecosystems spanning the period from 2008 to the present. It covers analysis of the general Danish telemedicine ecosystem, work on a new technological infrastructure platform, named Net4Care [47], and design as well as partial evaluation of a new organization created to accelerate the evolution of the ecosystem based on the Net4Care platform. The new organization is called "Stiftelsen for Softwarebaserede SundhedsServices" (in English: The Foundation for Software-based Healthcare Services), and hereafter referred to by its acronym "4S".

The 4S organization must handle a large set of diverse stakeholders, including national healthcare agencies, regional hospitals, software development houses, IT departments, etc. with highly complex interactions between. To understand and manage this complexity, we propose the concept of *software ecosystem architecture*, extending previous work by Manikas and Hansen [42]. We use this concept to frame a case study of the current Danish telemedicine ecosystem to inform the creation of 4S. In this context, the concept provides a common terminology across the central three structures of a software ecosystem: the organizational structure, the business structure, and the software structure.

We investigate this concept to ultimately answer the following research question:

> *"How can software ecosystems be modeled in a systematic way that allows reasoning about software ecosystems while details of the software ecosystem can be abstracted away?"*

Section 3 discusses the research question in detail.

Our contributions are threefold. First, we define and discuss the *software ecosystem architecture* concept. Secondly, we present the case of the current Danish telemedicine ecosystem in terms of the proposed concept, and discuss challenges that are relevant in areas beyond telemedicine. Finally, we present how the practice of software engineering is affected, through describing the creation and evolution of a central ecosystem architecture, Net4Care, that serves as a reference architecture and learning vehicle for telemedicine for the actors in 4S.

2. Related work

In this article, we analyze software ecosystems using the concept of 'software ecosystem architecture' and report on the case of the Danish telemedicine ecosystem. To our knowledge there is no previous work on software ecosystems for telemedicine or even for healthcare as such, apart from our own previous publications [24,15,16,40]. However, there is significant work on conceptualizing and modeling software ecosystems.

For example, there is related work that has influenced the way software ecosystems are modeled either by contributing to conceptualization [55,1,2,13] or by addressing a specific aspect of software ecosystems such as the platform and software component architecture [7,9,8,53,34].

Jansen et al. [33,31] and Boucharas et al. [12] propose the analysis and modeling of a software ecosystem from the software vendor perspective and separate ecosystems in three levels: the software ecosystem, the software supply network and the software vendor level. Jansen et al. [30], similarly, define three scope levels for software ecosystems: an external view, an internal view and an organization centric-perspective. Bosch [6] categorize software ecosystems according to their platform as operating system-centric, application-centric, or end-user programming-centric. Campbell and Ahmed [14] identify three dimensions of the engineering process of software ecosystems: business, architectural, and social. These map to our business, software, and organizational structure (with differences in the social dimension versus what the organizational structure covers). While their focus is on the engineering process, our focus is on the structure of an underlying software ecosystem.

Additionally, there is significant related work on quality aspects of software ecosystems. One such aspect is 'health' or 'sustainability'. In this context, van den Berk et al. [61] propose an ecosystem-based model for assessing the strategy of a software ecosystem called SECO-SAM. In their paper, they make an analogy between human health and ecosystem health and model the ecosystem health as being influenced by the biology of the ecosystem, the lifestyle, the environment, and the intervention of so-called healthcare organizations. Jansen et al. [30] define ecosystem health as a characteristic of the software supply network level in their three-level model mentioned above. Additionally, they propose the application of the measures of den Hartigh et al. [19] for defining the health of software ecosystems. van Angeren et al. [60] describe the robustness of the Iansiti and Levien [27] health measures of business ecosystems as an important factor for vendors that choose to depend on a software ecosystem. McGregor [43] translates the measures by Iansiti and Levien [27] to measures that can be applied to open source projects, Kilamo et al. [35,36] propose a framework for going from a proprietary to a Free/Libre/Open Source Software (FLOSS) ecosystem. One of the framework activities is setting up a "community watchdog" to assess three aspects of the newly created ecosystem: the community, the software, and "how well the objectives of the company are met". Although not directly stated, the watchdog indirectly assesses the health, while provide a number of measures to be applied in FLOSS ecosystems. Manikas and Hansen [41] propose a framework for the measurement of the ecosystem health. This framework consists of three ecosystem aspects that influence the ecosystem health: the actors, software and orchestration. In our work, we seek to understand how the structure of software ecosystems can be used to reason about these aspect also eventually with respect to health.

3. Method

We base our research on our own systematic literature review [42] and existing mapping studies and literature analyzes [1,55] to provide input and formulate our research question. The main research of this work can be summarized by the question:

> *"How can software ecosystems be modeled in a systematic way that allows reasoning about software ecosystems while details of the software ecosystem can be abstracted away?"*

More specifically, in this article, we are concerned with the following sub questions:

"(a) How can the concept of 'software ecosystem architecture' be used to model an existing (Danish telemedicine) software ecosystem to find areas of improvement?"

"(b) How can the concept of 'software ecosystem architecture' support the engineering of a new (Danish telemedicine) software ecosystem?"

In this article, we introduce the concept of *software ecosystem architecture* (explained in Section 4) and propose to model software ecosystems through their architecture. We apply our concept in a mixed method study: a descriptive case study in the Danish telemedicine ecosystem and an experiment in the Danish telemedicine ecosystem by creating the 4S organization including the Net4Care technological platform.

We use the Danish telemedicine ecosystem as our case study unit of analysis because we have access to information that can provide a deep insight on the case making it what Yin [67] refers to as a *revelatory* case study.

Our experiment focuses on the design of software ecosystems. We apply the concept of software ecosystem architecture and engineer a technological platform, Net4Care, investigating how a software framework can be engineered so that it can serve as a common technological platform, create the 4S organization to serve as the orchestrator of the platform, and change the business model of the telemedicine ecosystem.

Table 1 shows the data collection methods for the case study and the data created from our experiment. For both, the telemedicine ecosystem is analyzed in the three structures of the ecosystem architecture: organizational, software and business structure.

In order to describe the organizational structure of our case study, we collected data with qualitative methods, e.g., interviews/discussions with ecosystem orchestrators actors, external participants and interested parties from public authorities and organizations as well as companies. The public authorities included The National eHealth Authority, where we had several meetings with head of section, software architects and people responsible for standards: MedCom, The Danish Healthcare Datanetwork, where we held meetings with the deputy head and the person in charge of international projects; the five Danish regions, where we had several meetings with managers of telemedicine projects, people heading telemedicine centers, IT departments and departments for procurement and medical technology; and a small number of municipalities where we had meetings with managers of telemedicine projects and managers of departments responsible for telemedicine. From companies we had meetings with CEOs and telemedicine managers from providers of telemedicine, electronic patient records, and care records. The companies involved are Capgemini (now Capgemini Sogeti), CSC Scandihealth, KMD, Logica (now CGI), Systematic, SilverBullet, Sekoia, TDC, and Trifork.

In addition we studied publicly available records form several telemedicine projects, including the two largest projects from the National action plan for deployment of telemedicine. The records included meeting minutes, project reports and project participants.

Finally, we conducted a quantitative analysis of the network of actors and telemedicine application of a part (the largest of the five healthcare regions, the Capital Region, in Denmark) of the ecosystem [40].

Similarly for the characterization of the software structure of the ecosystem: we reviewed a number of existing applications and systems (FMK [22], RRS [54], TELEKAT [57], VAGUS, Telesår[58], EgenJournal [21], Sundt Hjem) and standards ([26], CDA ([5]), PHMR [52], XDS.b [66]), used the application network in the actor – application qualitative study mentioned above and interviewed an SMB in the field of telemedical application development Viewcare [62].

The variability of sources in the organizational structure and software structure of the case study addresses source triangulation [51,20]. Additionally, in this article we are using a mixed method of a case study and an experiment. Therefore, we obtain method triangulation [51,20] in the characterization of the software ecosystem architecture concept.

4. Software ecosystem architecture

We define the concept of 'software ecosystem architecture' by generalizing the definition by Bass et al. [4] of 'software architecture' (and extending on the definition of Manikas and Hansen [42]):

> The architecture of a software ecosystem is the set of structures needed to reason about the software ecosystem, which comprise actor and software elements, relations among them, and their properties.

The definition stresses that the architecture of a software ecosystem consists of multiple structures, each consisting of actor and software elements. Software forms the core of a software ecosystem, therefore the *software structure* of the software ecosystem is important. Moreover, a main purpose of software ecosystem actors is to create value (in a for-profit or non-profit manner) and thus the *business structure* of a software ecosystem becomes relevant. Finally, it is important to govern the interaction and organization of actors and software (e.g., for an actor to provide a software-based service in the ecosystem) and thus the *organizational structure* of a software ecosystem becomes important.

While many other structures can be discerned for software ecosystems, we argue that the above three are highly relevant and critical for reasoning about a software ecosystem. Consequently, we discuss these in turn and apply them in our analysis and design. Table 2 summarizes this discussion.

4.1. Organizational structure

The organizational structure of a software ecosystem contains actor and software elements that are related to the governance of the interaction and organization of the elements in the ecosystem. Important aspects of the organizational structure are the sets of actor and software elements included, the boundary of the

Table 1
Data collected and created during our case study and experiment.

Danish telemedicine ecosystem		Experiment: 4S	
Unit of analysis	Data collection	Unit of design	Data creation
Organizational structure	Interviews, publicly available records, qualitative analysis of ecosystem actor-application network	Organizational structure	4S organization creation
Business structure	Archival records of projects including Telesår	Business structure	Business modeling
Software structure	Interview with SMBs; analysis of existing applications, technologies, & standards	Software structure	Net4Care platform development & evaluation

Table 2
Examples of structures of software ecosystems.

Structure	Elements	Relations	Models
Organizational structure	Applications, platform, orchestrator, users, developers, boards, development projects, plans	Governed by, developed by, maintained by, connected to	Organizational structure, organization relationship & interaction, organization roles
Business structure	Products, services, partners, customers, resources	Channels, customer relationship, revenue stream	Business model canvas
Software structure	Modules, functions, services, nodes, developers	Depends on, used by, deployed on, developed by	Software architecture description

ecosystem they define, and how the structure governs interactions and support coordination among actors and software elements.

The interaction of actors is also related to the role each actor serves in the ecosystem. Specific roles might be more prone or even necessary to have interaction in different kinds of ecosystems, e.g., software developing organizations would have to interact with certification organizations in an ecosystem to promote their software products.

Moreover, the number of actors involved in an ecosystem and the level of selection they have to go through to be involved is part of the organizational structure. This is described by assessing how open the ecosystem is to external actors (e.g., by using the approach of Jansen et al. [32] or Manikas and Hansen [40]). Finally, the actors involved in the ecosystem usually have a commitment to the ecosystem. This commitment makes them aligned with the common goal of the ecosystem: its sustainability. Each actor has its own set of goals, however in order for the individual actors goals to be achieved and continued, the survival and thriving of the ecosystem is usually required. Exceptions are e.g. the cases where an actor decides to favor another ecosystem or where conflicts among actors weaken the sustainability of an ecosystem.

4.2. Business structure

The business structure contains actor and software elements that are related to how actors create, deliver, and capture value. 'Value' here refers to the benefit an actor gets from the software ecosystem, e.g., in form of need satisfaction or problem solution. A software element may deliver value in itself (e.g., an application) or be a resource in creating value (e.g., a platform or a reusable component/service). This structure is important in reasoning about cost, revenue, and/or sustainability of the software ecosystem. Note that business models do not necessarily model commercial organizations' businesses.

We describe business structure through business models using the 'business model ontology' formalism [49,50]. 'Business models' are here defined as

> A business model describes the rationale of how an organization creates, delivers, and captures value.

An excerpt of the business model ontology is shown in Fig. 1. Four aspects of business models are distinguished: the 'product', 'customer interface', 'infrastructure management', and 'financial' aspects.

In our models, we use the practical realization of Osterwalder's business ontology as 'business model canvases'. Here, the product aspects are covered by the 'value proposition' building block that concerns what values an organization is providing for a customer. The customer interface aspects are covered by the 'customer segment', 'channel', and 'customer relationship' building blocks that respectively describe types of customers, how customers are reached, and the type of relationships that are established to customers. The infrastructure management aspect is managed by the 'key partners', 'key resources', and 'key activities' building blocks. Key resources are the most important assets (e.g., physical or intellectual) whereas key activities are the most import actions (e.g., development or platform management) that are required for the business model to work. Finally, the financial aspects are covered by the 'cost structure' and 'revenue stream' building blocks that describe expense and income respectively in the business model.

4.3. Software structure

The software structure contains actor and software elements that are related to the production of applications in the software ecosystem. The primary actors are developers of the software ecosystem platform and of applications. Software elements may (recursively) be seen as consisting of multiple structures such as units of code (i.e., modules), runtime functions (i.e., components),

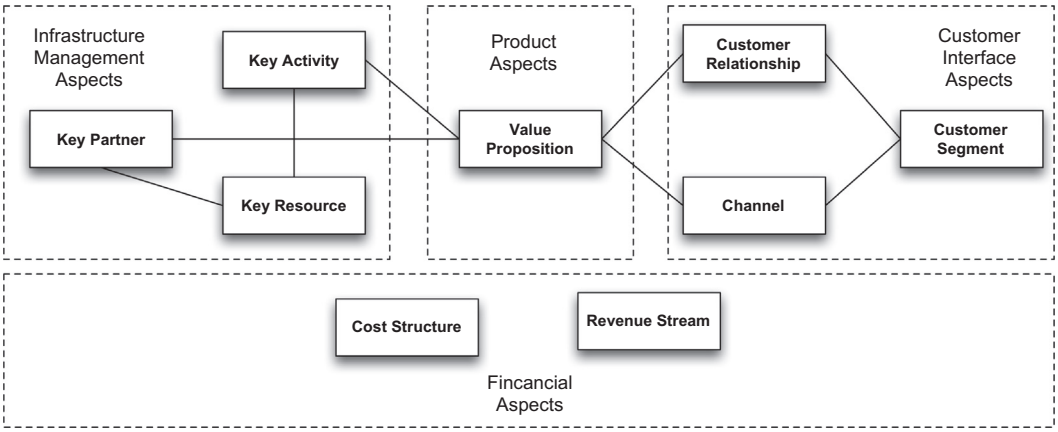

Fig. 1. Business model ontology taken from Osterwalder [49] with business model canvas terminology added.

or deployment nodes. These structures are important in reasoning about system quality attributes such as modifiability, performance, availability, and security [4]. Software architecture (models) are suitable for this type of reasoning and thus we model the software structure of software ecosystem architectures through software architecture descriptions.

We describe software structure in alignment with the ISO/IEC 42010 standard [28], see Fig. 2. Here an 'architecture description' consists of a set of 'architectural views' each of which adheres to the convention of an 'architectural viewpoint'. An example of an architectural description that adheres to this would be a 4 + 1 views-based architectural description [38]. Furthermore architecture decisions are described through an 'architecture rationale' and relations between elements in an architectural description are modeled through 'correspondences'. Architectural decisions are made to support 'architectural requirements'. An architectural view embodies 'architectural models' often in the form of (UML) diagrams. A view models one or more structures through its architectural models.

In our architectural descriptions, e.g., we use a set of viewpoints that, using UML, model how software is developed (through a "development view"), how software behaves at runtime (through a "functional view"), and how software is deployed to hardware (through a "deployment view") [25]. We use *quality attribute scenarios*, as defined by [4], to describe architectural requirements.

A example functional model for Net4Care (which will be introduced in Section 6) is shown in Fig. 8 on page 27, and examples of quality attribute scenarios are shown in Section 6.3.

4.4. Relationships among structures

The main relationships among the three structures that we emphasize are shown in Fig. 3. In addition numerous other relations exist. For example, a set of software services that are part of the software structure may provide the basis for the value creation of a business as described in the business structure and be created and governed as described in the organizational structure.

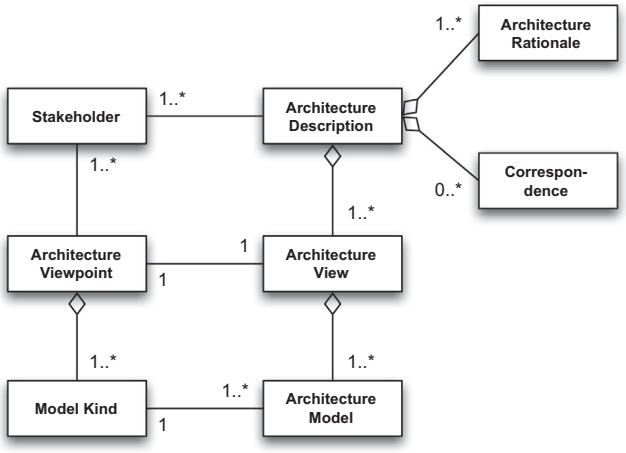

Fig. 2. Excerpt of the ISO/IEC 42010 architecture description ontology.

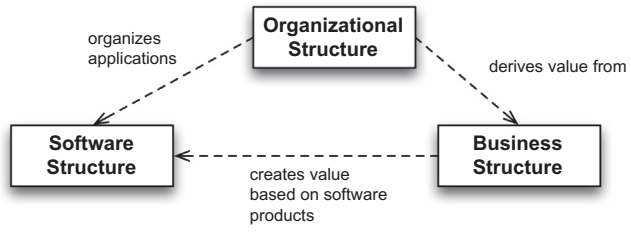

Fig. 3. Main relationships between structures.

5. Danish telemedicine software ecosystem architecture analysis

This study focuses on *telemedicine* as an application domain for software ecosystems in the Danish healthcare. The World Health Organization (WHO) defines telemedicine as

> The delivery of health care services, where distance is a critical factor, by all health care professionals using information and communication technologies for the exchange of valid information for diagnosis, treatment and prevention of disease and injuries, research and evaluation, and for the continuing education of health care providers, all in the interests of advancing the health of individuals and their communities [65].

To this extent, the unit of analysis of our case study is defined by the ecosystem around the telemedicine services in Danish healthcare. In this section, we describe our case study using the concept of software ecosystem architecture, described in Section 4. The aim is to find areas of improvements, cf. question (a) in Section 3. How we address the areas identified is described in Section 6.

Below, we first analyze the organizational structure of telemedicine in Denmark, the government policies developed to address need of increased uptake, and the issues the policy is intended to address. Next, we provide a more detailed analysis of current issues based on the business and software structures of the software ecosystem architecture. Finally, we outline the challenges remaining in the ecosystem under study.

5.1. Organizational structure

In Denmark, telemedicine services form part of the healthcare services offered by the public healthcare system. The telemedicine ecosystem is administered by the same administrative organs as the general healthcare. The organization of the healthcare system is fairly decentralised and has three administrative levels [48]:

State level The Ministry of Health governs the regional and municipal organization and management of healthcare. This includes organizations dealing with national infrastructure standards, architecture, compliance testing and implementation, and organizations dealing with cost structure and value creation.

Regional level Five geographical regions each own and run hospitals and finance private practitioners (in particular general practitioners (GPs)). Financing is based on transfers from the state and municipal levels. This includes organizations dealing with regional hospital systems: architecture, tenders and requirements, operation and national reporting and organizations dealing with general practitioners.

Local level There are 98 municipalities that are locally responsible for prevention, health promotion and rehabilitation. Furthermore, the local level is also responsible for health-related services such as home care and care in nursing homes. This includes organizations dealing with municipality healthcare: architecture principles, tenders and requirements, and operation.

A main issue is the relations among the different elements listed above, and the kind of coordination and predictability that these relations support. As a result of this, telemedicine in Denmark has been characterized by hundreds of uncoordinated, small projects, each developing their own solution, including infrastructure. Those solutions are not able to share data, due to the lack of common infrastructure, and they most often disappear when the resources of the project are consumed. This has resulted in more than 350 current telemedicine initiatives [44] of which the minority are in production.

This is also supported by our qualitative study of telemedicine applications and the organizations [40]. We analyzed the telemedicine applications and organizations for the Capital Region of Denmark. This is the largest, in terms of population, of the five healthcare regions in Denmark, with around 30% of the country's population. We identified the organizations related to the telemedicine applications implemented in the region and mapped the relationships between organizations, between organizations and applications, and between applications and applications. Our results revealed an ecosystem clustered around the telemedicine applications with low connectivity between the clusters. In other words, we noted a low application interaction and the tendency of the organizations to be connected to mainly one telemedicine application.

In order to examine how open the ecosystem is to external organizations, we separate the organizations in three roles that are important in the telemedicine ecosystem[1]:

- *Developing or external organization*. Organizations that are involved in the ecosystem with a specific task. This can include the development, maintenance or support of an application, the project management, supplier of related assets or services. The involvement of developing organizations is done, as we discuss in Section 5.2, either with a call for tenders where the actor with the accepted tender is appointed for the required task, or for services with cost lower than 500,000 Danish crowns (about 65,000 euros) direct fee-specific contracting. The ecosystem is relatively closed to external developing organizations: the ecosystem allows new developing organizations to be involved but new organizations are subjected to an acceptance rate (usually one out the applicants) and following a procedure of submitting a tender that might prove time and resource demanding. While direct contracting is sometimes allowed, this only includes minor tasks and the involvement is time-limited.
- *Host*. Organizations that are hosting telemedicine applications in their organizational premisses. This kind of organizations typically represent the product owner and customer/end user. The ecosystem is closed to new host organizations: There is currently a number of hospitals and municipalities that can be used as hosts in telemedical projects. Orchestrators can decide to include other organizations as hosts if there is a need. Moreover, in some cases, as we also discuss in Section 5.3, a developing organization might also serve as an application host. In that case, the ecosystem is almost as open to hosts of this kind as to developing organizations, but with the additional restriction that the host should commit to privacy and security regulations concerning healthcare data.[2]
- *Orchestrator*. Organizations involved in the governing body of the ecosystem typically responsible for the technological platform(s). As the ecosystem does not have one common technological platform, the assessment of how open the ecosystem is to organizations of this role is not possible. Orchestrators of existing platforms that are not ecosystem-wide (e.g. National Service Platform and Healthcare Datanet) have been introduced by appointment of state level organizations.

5.2. Business structure

Telemedicine systems, in a Danish context, are typically developed as part of a *project*. One or several orchestrators decide on investing in solving a specific problem. A project is then initiated where service provider actors (developing organizations) might be involved. The required activities and processes are identified and, if external actors are needed, a public call for tenders is announced. External actors are selected based on their proposals for large projects (cf. Section 5.1).

As an example, consider the national telemedicine project "Telesår".[3] The project aims at bringing expert diagnosis and treatment of ulcers to patients through the use of a mobile phone with a camera and a web-based electronic ulcer journal. The rationale behind this project is that expert ulcer diagnosis for patients with limited mobility (e.g., elderly) is expensive but at the same time necessary as these patients are in the high risk of developing severe ulcer complications often resulting in amputations. A usage scenario is that a home-care nurse visits an elderly patient with a diabetic foot ulcer and – in connection with changing the bandage – takes a picture with the mobile phone and then uploads it in the "ulcer journal". If the nurse finds that action might be needed immediately, she may set up an on-line conference with a dermatologist who, at the backend, logs in to the ulcer journal from his or her office and analyzes the picture of the patient, evaluating how the patient needs to be treated. If no immediate action is deemed necessary the dermatologist evaluates the pictures off-line.

In this scenario, the patient receives better diagnosis and treatment with fewer visits to the specialized hospital departments. The dermatologist can do a better job through closer observation and a specialized record. Furthermore the specialist may treat more patients. Also the home-care nurse learns from the dialog with the dermatologist and is thus able to provide the patient with better care. The regions and municipalities get more value for money, since faster diagnosis and earlier treatment may be provided, while the state increases the efficiency of the provided healthcare services by reducing costs.

A business model canvas for the "Telesår" project is shown in Fig. 4. The history of the project and the business model illustrates some of the issues with the current Danish telemedicine ecosystem. First, projects related to telemedical ulcer treatments have a history that goes back approximately 10 years as local (medical) research projects [17]. Secondly, the project, while now being implemented on a national scale as one of the five projects mentioned in Section 5.4, uses a proprietary electronic journal ("pleje.net") that to our knowledge does not integrate with national or international standards and that as such does not reuse national services (see Section 5.3).

5.3. Software structure

In 2010, we conducted a study of the technical status of current telemedicine projects (RRS, TELEKAT, Telesår, EgenJournal), a commercial system (ViewCare), and international experience and standards (HL7, CDA, and PHMR).

The study included both actors and software architecture elements and relationships, and included interviews with regional hospital IT departments, architects and developers, documentation

[1] A similar classification and evaluation was followed in Manikas and Hansen [40].

[2] Traditional hosts like hospitals are also committed to this kinds of regulations, but this does not appear as cumbersome as this is part of the everyday work in a hospital.

[3] http://medcom.dk/wm112455.

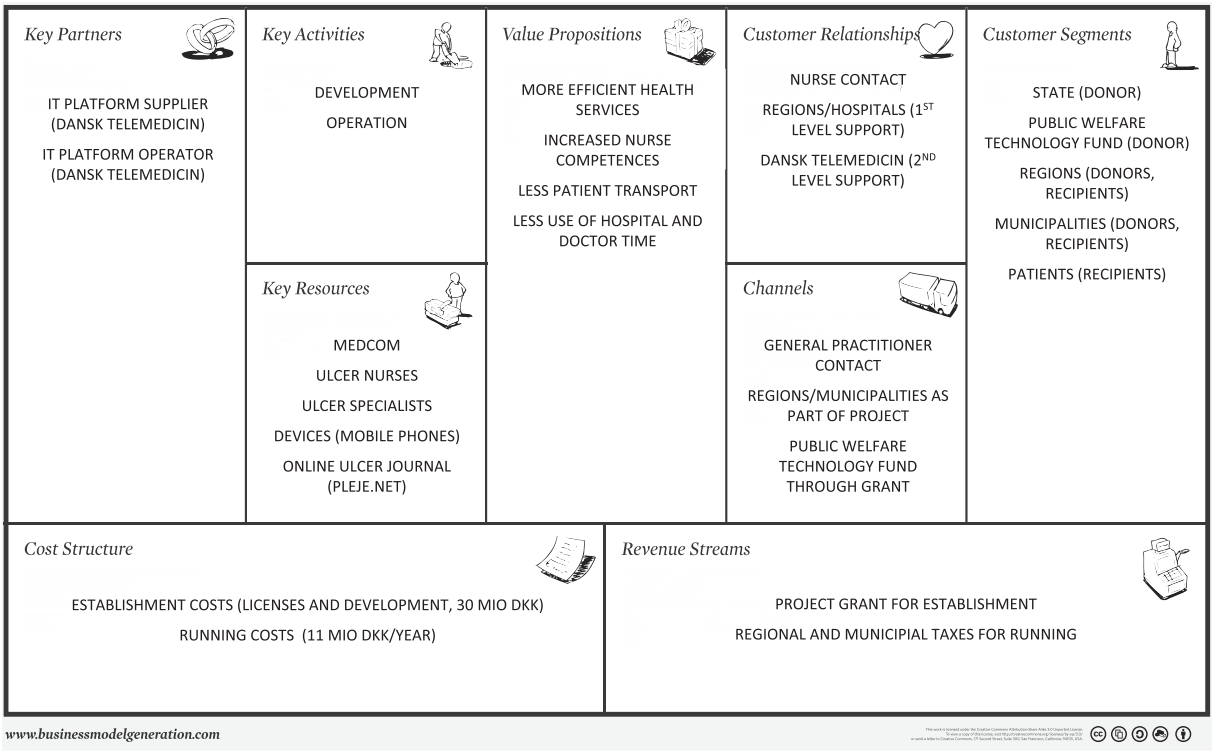

Fig. 4. Telesår telemedicine project organization business model canvas.

reviews, and the development of architectural functional and deployment views for the studied systems. Based on this raw material, a commonality/variability analysis was conducted.

Perhaps not surprisingly as all studied projects and products have the same core use case: "Measure clinical information in the home of a patient, send it to a hospital server for storage and review by a clinician", the same software architecture was shared between all software systems. They were all variants of a three-tier information systems, as depicted in Fig. 5. That is, a monitoring application deployed in the home collects various measurements and uploads them to a proprietary project server, stores data in a proprietary format in a proprietary database, and allows clinicians to browse and view data using a clinician application. Essentially all projects were "stove-pipe" systems as generally there were no reuse of software modules, of data formats, or of databases among projects. In other words, there was many software system but no ecosystem-wide platform. In addition, existing platforms were only used in a minority of the systems.

One explanation was the lack of an organizational structure in 2010 to govern a coordinated development effort between projects to ensure reuse and ensure a use of common formats and database infrastructures. The period was that of exploration and early experiments, often driven by local initiatives and funding.

Our study formed the basis for outlining architectural requirements for a framework/platform (Net4Care) that could serve as software structure for an ecosystem for telemedicine.

Specifically we identified three main issues, detailed in [16], in the current state that from a technical viewpoint inhibit a natural evolution of the ecosystem and that had to be addressed.

- *Lack of integration* among (tele-)medical systems. The systems are not integrated and often replicate data in diverse data formats and storage systems.
- *Missed opportunities for reuse*. All studied systems had essentially the same architecture, but all had built their own software modules, used proprietary data formats, and hosted own data centers.
- *Low buildability* of integrated telemedicine systems. Bass et al. [4] defines the architectural quality attribute buildability as: *Buildability . . . refers to the ease of constructing a desired system.*

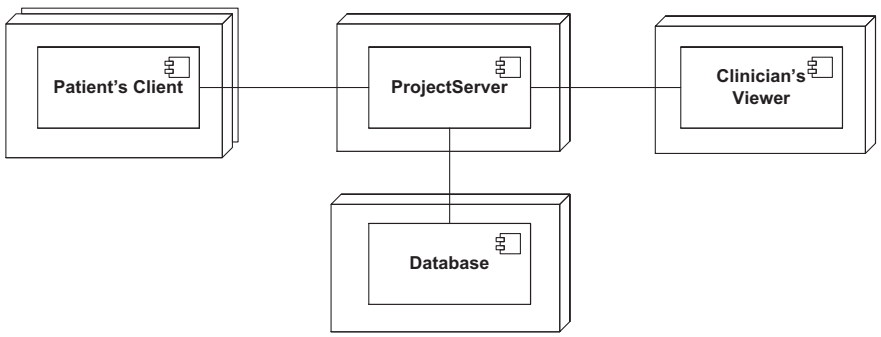

Fig. 5. Deployment view of typical telemedicine project.

The few systems that allowed integration used standards with very steep learning curves, there was a lack of tutorials and test environments, and it was closed source software, which made development prohibitory slow and expensive.

As an example of *lack of integration*, in a case of telemedical care of Implantable Cardio-Defibrillator (ICD) patients at Rigshospitalet in Denmark, doctors, and analysts had to consult up to 12 different applications at the hospital to cater for remote patients. Another example was a small SMB that had developed a system for *Chronic obstructive pulmonary disease (COPD)* patients that provides teleconferencing and digital measuring of spirometry and subsequent upload and review by the clinician. While successful from a clinical and patient point of view, the SMB hosted its own servers with the databases and the clinicians' system at the hospital had to be on the SMB's VPN. This lack of integration forced the clinician to copy and paste measured values manually into the hospital's EPR system. The SMB argument for this lack of integration with the regional hospital's infrastructure was mainly speed of deployment and maintenance and thus the overall perception of the company's solution.

Regarding *missed opportunities for reuse*, all studied systems has similar functionality (medical data upload, database storage and retrieval, browsing and review by clinicians, etc.) but all code modules were developed from scratch. The lack of a common, shared, infrastructure to reuse forced companies to build all components themselves. Thus, small to medium sized business with expertise in e.g. home care were essentially excluded for the market due to lack of resources to develop and host server side solutions.

As examples of *low buildability*, a PhD student in the Net4Care project spent more than 80 hours demonstrating that medicine data could be read for a telemedicine application from a national web service built on top of the National Service Platform (NSP) for healthcare integration [59]. Several sources contributed to the slow process. While documentation of the web service interface existed, it was not detailed enough to allow a valid SOAP message to be composed, the WSDL contained defects that made it impossible to generate client stubs, descriptive error codes from the service were missing making defect tracing cumbersome, and the security model code assumed non-existing certificates and provided yet another layer of required understanding. Moreover, no "server in a box" existed so learning and test programs depended upon test servers outside our control and whose characteristics were changed without notification making the developed software fragile.

The three issues are of course not orthogonal: increased reuse leads to faster and more reliable development and thus to improved buildability, and reusing storage systems and standardized data formats leads to easier integration. The issues, however, works strongly against a natural emergence and evolution of a software ecosystem. Without common and shared software architectural standards, data formats, storage systems, modules and information resources, development and integration is difficult, especially for small and medium sized business.

These issues must thus influence the software engineering practice and shape how a software ecosystem is created and evolved. The Net4Care ecosystem framework was specifically designed to address these issues, as detailed in Section 6.3 below.

5.4. The challenges that remain

To address issues in the telemedicine ecosystem, the government set up a task force in cooperation with the major actors among the ministries, the Danish Regions and Local Government Denmark ("Kommunernes Landsforening"). The aim was to develop an action plan for the deployment of telemedicine [23].

The plan outlines five national, large scale telemedicine projects, and in addition mentions the development of a national reference architecture for telemedicine [46]. The projects are intended to develop and deploy software, including some infrastructure elements and to provide experience with and evaluation of the software and organizational set-ups used. At the end of the experiments, the successful parts are to be deployed nationally.

The national plan is primarily intended to address two important issues: scale and quality. First of all, telemedicine solutions are now likely to be deployed on a national scale over the next five to ten years. Secondly, the quality of the solutions is likely to improve both with respect to software and to organization. This likely quality improvement is due to the fact that many actors will share the cost of developing and maintaining the software. Furthermore it will be easier for the actors to share experience on organization.

While the action plan tackles issues that need to be addressed, our analysis[4] points to some important concerns that should be addressed if the ecosystem is to grow and attract numerous players during the next years.

In short, the current ecosystem is suffering from weak organizational, business and software structures, and very weak links between them. This results in high transaction costs summarized by the following: ad hoc solutions to coordination and development (weak organizational structure and weak links), low attractiveness of markets (weak business structure not being addressed by the organizational structure) and low accessibility (weak software structure not being addressed by the organizational structure). Below we consider these in turn.

Ad hoc solutions: The current organizational structure is not able to effectively develop the software structure and the business structure as described in Sections 5.1 and 5.2. When major problems are identified ad hoc solutions, like the task force developing the action plan, are chosen and these solutions usually address problems in the software structure directly instead of developing the organizational structure to create effective links between the structures.

Low attractiveness of markets: Current markets for telemedicine in Denmark are fragmented and their characteristics are very difficult to identify and they are changing. Thus there are no credible long term plans for how to develop telemedicine at the regional level. The closest to such plans are the government plan for deployment of telemedicine and the Danish Regions Shared Indicators for the digitization of healthcare [18]. As described above the government plan mentions infrastructure elements and a five telemedicine projects under development and evaluation. The indicators of the regions list four of the five projects of the government plan. Thus, from a business as well as a hospital point of view, the current plans imply that the markets are more or less on hold until the experiments are finished and evaluated. At that point in time, if the projects are successful, a small number of telemedicine systems together with some infrastructure elements will be deployed nationally. If some of the experiments fail nothing is stated about what will happen. In neither case do the current plans provide a way forward for companies that want to market telemedicine systems in Denmark, except for the companies providing the systems used in the experiments.

The municipalities are several steps behind compared to the government and the regions due to the fact that up until now mainly hospital departments have driven the numerous telemedicine projects in Denmark. In a recent policy paper on telehealth [37], the Danish Municipalities outline a strategy and point to

[4] This analysis is done as part of the national project "Denmark as a telemedicine pioneer" and based on interviews and workshops with more than 50 companies.

some of the challenges facing the municipalities, including the need to supplement the "hospital/diagnosis/illness-specific" perspective with a "non-illness specific" perspective. In addition, the challenges include problems with existing economic models and the need for new ways to share costs between region and municipalities, poor equipment and system quality, and lack of system integration. In the long run the new focus on telemedicine and telehealth by the municipalities will almost certainly strengthen the development of the ecosystem. However, in the short term the most likely effect is that municipalities will do some more preparatory work before investing in telemedicine and -health. In summary the current business structure is weak and the ad hoc solutions at the level of the organizational structure do not create links that are capable of improving the development of the business structure.

Low accessibility: The companies that do decide to enter the market, and their customers, are faced with another challenge: the low accessibility of public healthcare infrastructure and services. This low accessibility is due to several factors: written documentation is sparse and not of high quality, often the rationale behind solutions is not obvious and no explanations available, and there is no support in terms of tutorials or "help desks". The result is that the learning curve for a company that wants to use the Health Data Network and/or some of the associated services like Shared Medicine Card[5] is very steep and often prevents SMBs from entering the market. In summary the current software structure is weak and the ad hoc solutions at the level of the organizational structure do not create links that are capable of improving the development of the software structure.

6. Danish telemedicine software ecosystem architecture design and realization

In the following section, we describe our experiment using the software ecosystem architecture concept introduced in Section 4.

During 2008, we came to realize the need for significant improvements of the Danish telemedicine ecosystem, c.f. Section 5.1. Our first attempts on setting up research and development projects to address the situation focused on two issues:

- Improving the software structure: we began to work on national infrastructure based on international standards and supporting integration between systems from different vendors, as well as increased international market potential.
- Improving the business structure: we tried to develop business models for different types of participants, e.g., SMBs with telemedicine products, providers of hardware, infrastructure companies e.g. telecoms, and the different providers of telemedicine services, including hospitals and municipalities.

As mentioned above and described below in more detail, our work on improving the software structure through national infrastructure based on international standards progressed well and we developed an effective set of tools and tutorials as the first elements of the technological platform of the ecosystem. We named the platform "Net4Care", after the project that provided most of the funding.

However, it turned out to be more difficult to create convincing results regarding improvement of the business structure, i.e. what would the benefit be for e.g. an SMB developing telemedicine products? A main challenge was that the links between the software structure, the organizational structure and the business structure were weak. Thus the impact of positive developments in software structure was slow to develop in the business structure. Concretely, paths from our research in terms of the Net4Care platform to uptake was quite long and uncertain since we had no direct connection to national or regional forums making decisions in the area, i.e. to the relevant parts of the emerging organizational structure.

In order to improve the situation and accelerate the development of the ecosystem, we decided to supplement our work on the software structure (work on the Net4Care infrastructure platform) and the business structure (work on business models), with an effort directed towards the organizational structure. Thus, we began to work on establishing a new open source foundation with the mandate to promote telemedicine and representing relevant, interested parties. The intention was that this new organization, 4S, should play a key role in the acceleration of ecosystem development as an orchestrator in the organizational structure.

6.1. Organizational structure

The new 4S organization should be an orchestrator responsible for the open source infrastructure platform for telemedicine, Net4Care (see Section 6.3) and have the qualities to maximize the likelihood of success. We made literature surveys on open source, healthcare, and software ecosystems [42] and studied involved Danish organizations and their plans, including public open source organizations. Based on this, we decided upon a set of characteristics of 4S. The characteristics covered both organizational structure aspects, such as the mission and structure of 4S and community building (including with healthcare professionals and patients), aspects related to business structure, e.g. developing viable business models, and aspects related to software structure, e.g. understandability and accessibility of software resources, and "traditional" open source organizational aspects, e.g., governance. During 2012, we then held a number of meetings and workshops with relevant, interested parties where we presented the ideas and the rationale behind 4S. The status after the initial round of discussions were:

Mission: There was general agreement on the need for an organization that could play the role of the orchestrator in the organizational structure and streamline the ecosystem around an infrastructure framework like Net4Care. The main critique concerned the likelihood of sufficient backing from major players and speed of uptake.

Structure of the organization: In order to speed up the formation of 4S and to create an agile, adaptable organization, we proposed a bottom-up approach where interested individuals from relevant organizations at the state, regional, and municipality levels agreed upon a proposal and then tried to secure organizational backing and membership. This in turn led to a structure where we could allow only a few organizations to be mandatory members from the start. On the other hand informal accept from most was important. There was general agreement that it would not be feasible to establish 4S through a process where all the major bodies at the state, regional, and municipality levels as well as business representatives were invited. There was some concern about the likelihood of convincing the needed mandatory members to participate.

Community building: Current Danish telemedicine initiatives are mainly controlled by IT managers and project leaders. This in turn often leads to dissatisfaction among the involved healthcare professionals and less than optimal results for the patients. To change this we proposed to create a number of healthcare communities as an important part of 4S. There was general agreement on—and only few discussions of—these issues. However, we ourselves became concerned about how to create this involvement, since the main elements in the technological platform of the ecosystem, i.e. the Net4Care framework, was about infrastructure. Thus, the platform

[5] https://fmk-online.dk.

itself did not seem to be a good vehicle for discussions of challenges in telemedical treatments of different types of illness.

Traditional open source organizational aspects: This concerned primarily how to create developer involvement and quality assurance. However, we also considered development plans/projects and their creation and prioritization to be very important. Especially we wanted to secure the influence of the different communities.

Understandability and accessibility: In our view, 4S had to provide a very high degree of understandability, accessibility, testability etc. of the elements of the technological platform in order to become a success. We already had some very good results in this area through the Net4Care website[6] and there was general agreement on this. Especially commercial companies could list numerous examples on non-understandability, non-accessibility and lack of test environments concerning existing healthcare infrastructure, cf. also Section 5.1 and Section 5.4.

Two important developments in the period from the end of 2012 until the late spring of 2013 changed the prospects for 4S in a very positive way. First, the Alexandra Institute and three other organizations were granted a large telemedicine contract by The National Board of Technology and Innovation with one of the authors as project leader. This provided resources for the work on establishing 4S, and – even more importantly – it gave new legitimacy to our efforts on making 4S an orchestrator in the Danish telemedical software ecosystem.

Secondly, a draft of the new national reference software architecture for telemedicine systems was published [46]. This reference architecture closely followed the architecture that we had worked with in Net4Care, and thus the platform was now viewed as an implementation of the reference architecture. The key elements of the platform (i.e., the Net4Care framework and test-environment) began to be used as a common technological platform in the software ecosystem: The framework and test environment was used as tools for both companies and healthcare service providers to evaluate and experiment with how different systems and components fit the reference architecture, how to modify them to increase compliance and how to design and implement new interface based on the standards of the reference architecture to increase data sharing.

These developments very directly impacted both the backing from major players, which were one of the outstanding issues, cf. "Mission" above, and the commitment from mandatory members, cf. "Structure of the organization" above. The only major outstanding issue was how to create viable communities.

The technological platform with the Net4Care framework and test environment provided a good basis for community building involving it people from both private companies supplying IT systems and from organizations delivering healthcare services, e.g. the regional IT departments hosting the IT systems of the hospitals in their region. However, it was not well-suited for organizing discussions around how to improve telemedicine for different groups of patients or citizens and for the healthcare professionals working with these groups. In other words, the platform, with its focus on the Net4Care framework and test-environment was not suited for this kind of community building. In order to advance the creation of active communities we decided to make software used directly by patients/citizens and healthcare professionals a primary concern of 4S.

One way to do this was to make agreements with a number of vendors supplying such systems and using the Net4Care framework and test-environment. We are working on this and expect it to be an area of activity that will grow steadily in the future. However, these telemedicine systems are closed source and setting up agreements with the vendors is a slow process. In addition most vendors considers experiments with and development of their systems as something that is company internal and under their tight control. In short, most vendors do not view their telemedicine systems as potential elements in a technological platform of a software ecosystem. And from a user perspective (patients/citizens and healthcare professionals), the path from a successful experiment to new versions of the commercial software to be improved via the experiments is very long and the conditions governing the company decisions are opaque.

Thus, we decided to pursue a different path as our main strategy: the inclusion of a major open source telemedicine platform as part of the technological platform for which 4S was responsible. We had for some time had discussions with the three regions responsible for those of the national large scale experiments that were developing infrastructure elements and a telemedicine system for handling data collection by the patients themselves, cf. Section 5.4. Backed by the alignment of the reference architecture with the Net4Care framework and test-environment as well as the legitimacy provided by the large telemedicine contract granted by The National Board of Technology and Innovation we managed to make an agreement with the three regions that the telemedicine platform in question, called Open Tele, would be handled by 4S. This agreement in turn made it attractive for the five groups of patients/healthcare professionals currently using systems based on the Open Tele platform to participate in communities organized by 4S.

6.2. Business structure

Fig. 6 shows the business model canvas for the 4S organization. The central value propositions are cheaper telemedicine IT development through ease of system integration and cross-sector and -supplier integration. This is to be achieved through platform development, (test) platform hosting, and ecosystem governance, the key resources include open source components (Net4Care and OpenTele) and integrated national healthcare services.

Fig. 7 shows the Telesår business model under the assumption that the Telesår project becomes part of the 4S ecosystem. This adds to the value propositions of the Telesår project in that IT development would arguably become cheaper (faster), result in a better integrated system, and be based on open standards. This should result in a changed cost structure in which establishment and running costs should be lowered (but a stakeholder fee to 4S should be added). As part of the Telesår project, there is now the potential that parts of the project can be made available as services/components to other telemedicine applications (see "Key Activities" and "Key Resources") with the possibility of use by others (see "Customer Segments" and "Revenue Streams").

6.3. Software structure

The 4S ecosystem's software structure elaborates on the Net4Care framework. This framework was architected based upon the three issues identified in current telemedicine: lack of integration, low buildability, and lack of reuse; and designed as best possible to fulfill the requirements of an *application-centric ecosystem* [6]. The other categories of ecosystems as defined by Bosch: operating system-centric and programming-centric; seemed less relevant for telemedicine. Bosch lists four essential success factors for an application-centric ecosystem:

1. The foremost success factor is a *large set of customers* or the promise of those customers.

[6] http://www.net4care.org.

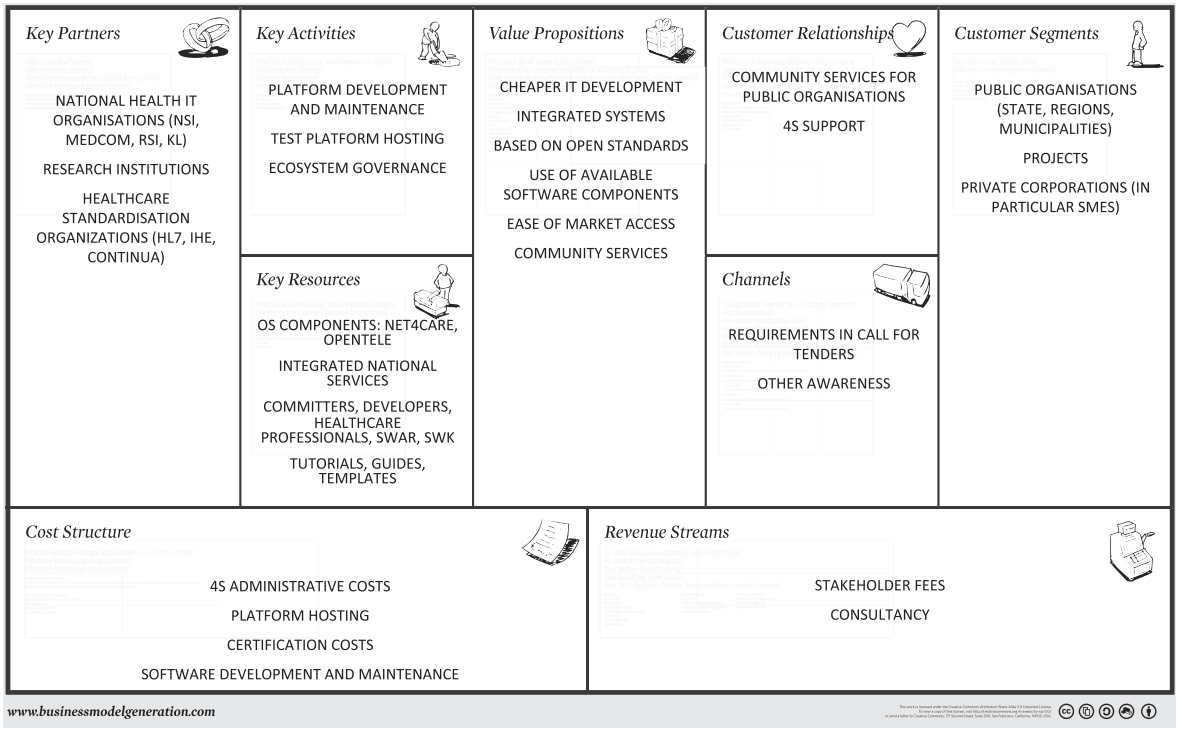

Fig. 6. 4S business model canvas.

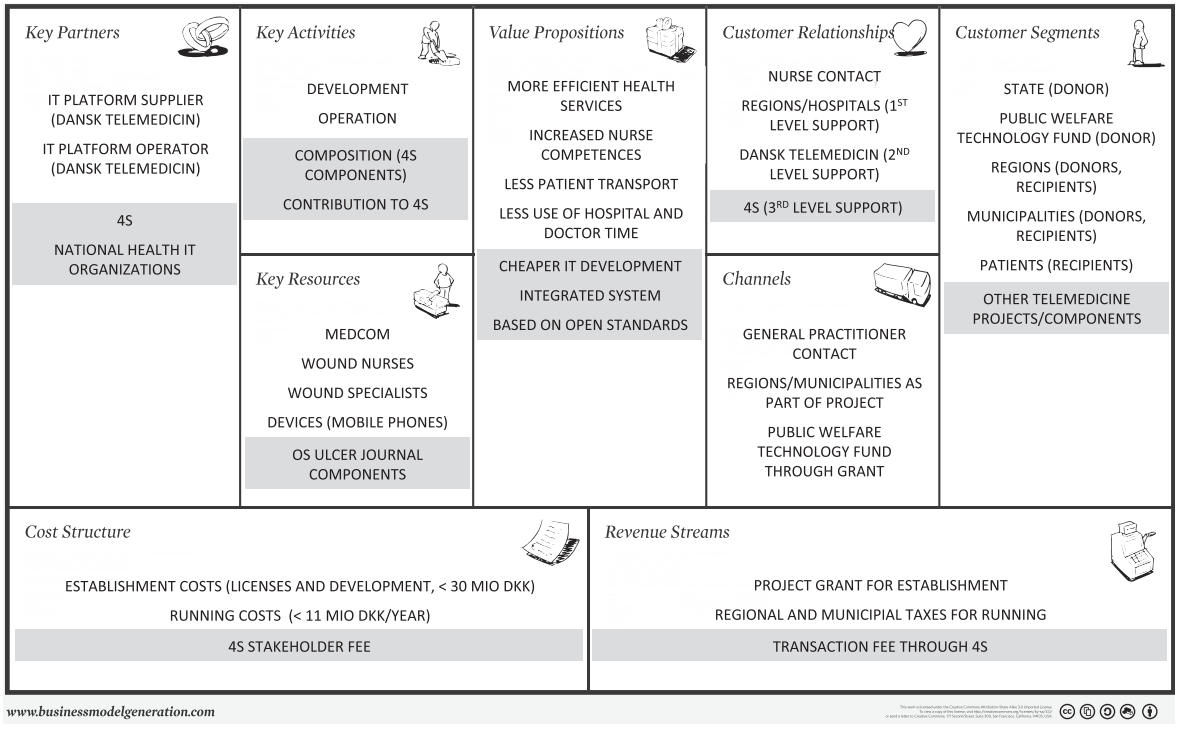

Fig. 7. Telesår and 4S business model canvas. Grayed blocks signify potential implications of 4S on the Telesår business model.

2. The platform company should *aim to simplify contribution by third party developers*, by popular development environments, stable and expressive interfaces, and easy deployment and integration.
3. The platform should provide solutions to *extend data models and work flows* as well as integrate into the same user experience.
4. The platform should provide a *viable sales channel* where third party contributions are exposed to customers.

Of these, points (2) and (3) are achievable by a strong commitment by the technical development team and realizable in the software structure, point (4) must be provided by the organizational

and business structure, and finally (1) can be hoped for from the quality of the efforts invested in all three structures.

Our analysis showed little focus had been on the simplification of contribution and extendability of data models in prevailing infrastructure and telemedicine pilot projects. Thus, our team set out to develop an architecture for the ecosystem platform with special attention to these aspects and inspired by the functional and architectural requirements elicited in the analysis phase.

The main architectural drivers became the following quality attribute scenarios [3]:

> *(QAS 1/Modifiability) An SMB developer with a strong background in electronics and hardware-near computing [source] wants to develop a telemedical application for the home that supports uploading measured clinical values [stimulus] using the Net4Care framework [artifact] as part of preliminary exploration and prototyping [environment]. The developer downloads, installs, and tests a first prototype having a full round-trip of clinical measurements (from device to simulated server and back again) [response] within four staff hours [response measure].*

This scenario captures an important aspect of Bosch's requirement of *simply contribution* namely the ability for third party to understand and develop using the framework.

> *(QAS 2/Integrability) A clinician [source] wants to review telemedical measurements of a patient [stimulus] using the regional Electronic Health Record (EHR) system [artifact] as part of daily work [environment]. The clinician can query, view, and annotate the data [response] without need of starting specialized telemedicine applications nor copy-pasting data between applications [response measure].*

This is another aspect of Bosch's second point, namely *easy deployment and integration*, in that the Net4Care platform seeks to avoid stove-pipe systems with its own clinician applications, databases and servers, and replace them with integration into existing clinical systems.

> *(QAS 3/Modifiability) An SMB developer [source] wants to support a new type of measurements in the home [stimulus] using the Net4Care framework [artifact] during development [environment]. The developer defines appropriate software modules with proper clinical encodings and integrates them into the Net4Care operational environment [response] within two weeks [response measure].*

This final architectural driving scenario captures the third Bosch requirement of *extend data models and work flows*. The scenario requires two parts. The first is that the technical framework contains adequate hotspots/variability points to allow customization and extending of data types, and the second is that there are certification processes in place that verify that the choices made concerning clinical encoding are clinically valid so the resulting data produced can integrate into EHR and other clinical systems.

6.3.1. Net4Care architecture and reference implementation

To support the main three architectural requirements in the form of the scenarios (as well as a set of architectural and functional requirements reported elsewhere [25]) we made the following central architectural decisions.

- *Information resources* primarily in the form of open source well proven and tested tutorials on the web site www.net4care.org provides a shallow learning curve for developers.
- *Reference implementation* as open source.
- *Staged testing environment* which provides a simple isolated test environment for fast development that seamlessly can be migrated into a full operations environment.
- *Clinical standards* used at the back tier for storage format (Continua Alliance *Personal Health Monitoring Record* (PHMR) [52] which is a HL7 "Clinical Document Architecture" standard [5]) as well as for database system (XDS.b *Cross-Enterprise Document Sharing* [66] which is a standard for clinical storage systems).

An overview of the functional run-time view of the Net4Care framework is shown in Fig. 8. We follow Bass et al. [3][§9]'s recommended practice to represent the component-connector view using UML object notation: Boxes represent run-time components, single line boxes are (passive) objects while boxes with an extra vertical line represents processes (active objects). Links represent connectors that mediate data and control flow at run-time between components. Verbs on the links describe the type or role of data/control exchanged.

Note that while object notation in its UML interpretation only represent one of potentially many different instances of a system, Fig. 8 represents the general configuration of run-time elements that all instances of Net4Care will have. For instance, any HomeClientApplication will have any number of StandardTeleObservations but only one DataUploader, etc.

In the figure, components marked by light gray are those the SMB developer must develop, while the rest are provided by the Net4Care framework. The web site's information resources provide detailed tutorials regarding download, installation, and first exposure ("Hello World") in using and configuring the framework. The home client framework is available in Java and C#, while the server is purely Java based.

The *SMB Comp* is our abstraction over all code that is not related to the Net4Care framework except for the fact that it must produce medical measurements, like blood pressure, weight, spirometry measurements, etc. Such measurements we denote *telemedical observations*. Thus it is in this component that the SMB will invest time and effort to have a competitive edge in the market for telemedicine: ease of use, appealing interface, intuitive device coupling, etc. Though it is represented by a single gray box, it should by far be the greatest investment by the SMB as Net4Care should ideally handle the rest.

The *ObservationSpecifics* is the framework's main variability point with regards to the concrete types of telemedical observations that the SMB wants to support. This requires some insight into clinical informatics, notably code systems which are used to uniquely identify the type of measurements made. Here, we have invested special attention to provide guides, tutorials, and code support that allows non-expert users of Net4Care to produce clinical valid data while shielding them as best possible from the full details of HL7 which has a very steep learning curve.

The *DataUploader* is a standard client side class that handles all the transport to the application and storage tiers. The Net4Care server is responsible for translating the core data from the client side into full and validated PHMR documents and store them in national/regional XDS.b. Once stored, clinician systems like EHR can retrieve data using standard XDS.b profiles.

The *ServerConnector* protocol between the client system and the Net4Care server and the *XDS.b Adapter* component in the server are also architectural variability points and serve the requirement of a staged testing environment. These can be configured by the SMB developer in stages, where the simplest uses local, simplified, variants which allows fast and non-distributed development. Progressively more complex variants are provided by the Net4Care platform, like an HTTP connection to a OSGi-based webserver, and ultimately the production ready variants using secure HTTPS connections and real XDS.b interfaces. These variability points thus support both QAS 1 and 3.

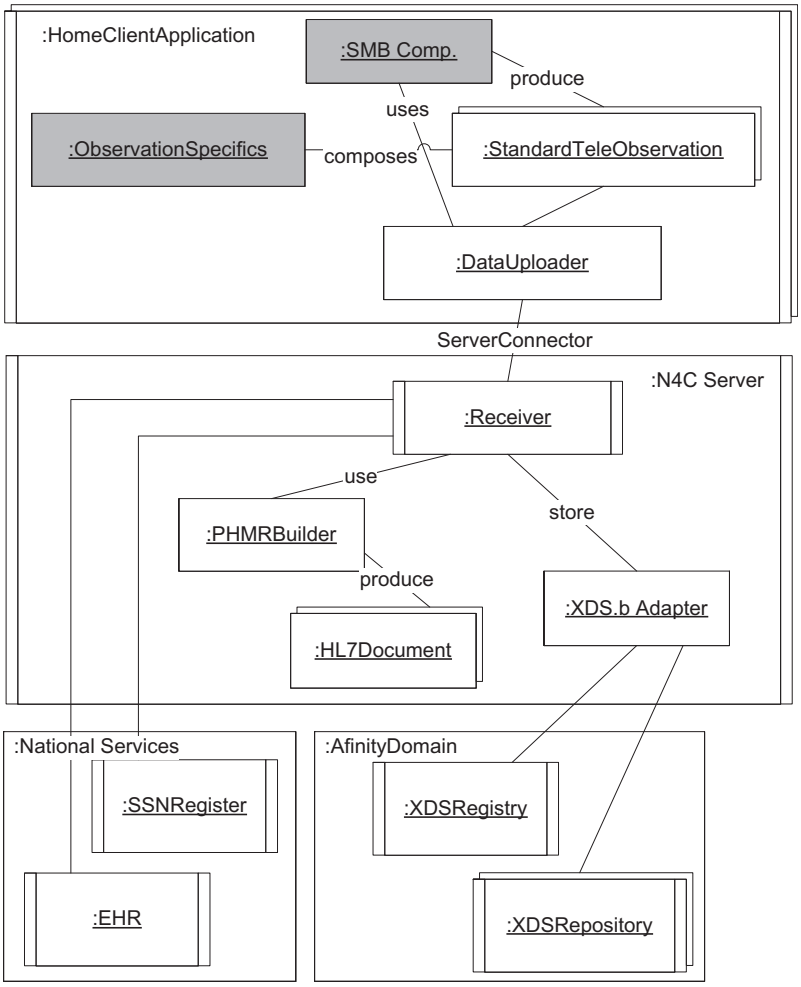

Fig. 8. Functional view of Net4Care. UML object notation is used to represent the run-time structure of any instance of the framework. The notation used is explained in detail in the text.

One lesson learned from the development of the software structure is that the engineering practice for software ecosystems must be aligned with lessons learned from analysis of successful application-centric ecosystems. Notably, the requirements of *simplified contribution by 3rd party* and *extensible data models and work flows* can be supported at the software framework level if included early in the architectural analysis. Moreover, these points must have strong focus in the creation and evolution of the ecosystem.

6.3.2. The Open Tele data collection telemedicine system

As described above, we decided to include an open source data collection system called Open Tele in addition to Net4Care. The idea was to use Open Tele to generate interest among healthcare professionals and citizens/patients and thus supplement the interest among IT people generated by Net4Care. Basically, Open Tele corresponds to the box "SMB Comp" in Fig. 8. The Open Tele system includes a tablet used by citizens/patients for data collection, including device that can be coupled to the tablet. In addition, the system provides local data storage and access for healthcare professionals.

The system is a new addition to the ecosystem and to 4S, and we do not yet have much experience with it. However, we expect it to play two important roles. First of all it will allow us to set up experiments together with healthcare professionals and citizens/patients where we are free to choose which parts of the system we want to work with. And, if successful, the results of an experiment may be included in the Open Tele software system if the necessary resources for implementation according to the 4S quality requirements can be provided. In addition, commercial vendors working with closed source are free to use the same results to improve their systems. Secondly, we plan to further develop Open Tele to work together with an app store approach to open a new channel for contributions.

In terms of the essential factors from Bosch, Open Tele contributes significantly to a large set of customers or the promise of those customers primarily through its enabling of community building. The app store approach will both simplify contribution by third party developers and provide a viable sales channel. Finally the possibilities of Open Tele in combination with Net4Care provides solutions to extend data models and work flows as described in Section 6.3.1 above.

7. Evaluation

The 4S software ecosystem is now being realized. Obviously the quality of the Net4Care framework and its compliance with the national reference architecture is a key element in this. But our discussions with actors and other stakeholders indicates that the establishment of 4S as a credible orchestrator has been, and is, crucial to the evolution of the ecosystem.

Our primary evaluation to date, however, has been through the use of the Net4Care framework. This framework has successfully

fulfilled the aspect of *"technological platform . . .that allows involvement and contribution of the different actors"* and *"with symbiotic relationships"* [42]. Table 3 shortly outlines involved actors and their main interests, which we will expand upon below.

The actors so far can coarsely be classified in three classes: private companies, regional hospitals and their healthcare IT departments, and educational institutions, each having their own interests and perspectives.

Sekoia [56] is a small company (with less than 20 employees) that develops IT solutions for healthcare with strong commitment to user-driven innovation to ensure high quality in use of their products. With this emphasis they are less inclined for large investments into back tier development such as hosting data warehouses and integration with existing clinical systems. Thus the Net4Care framework is well suited to handle back tier aspects and presently integration experiments with their Android tablet end-user solutions is explored.

The interest from the regional healthcare IT departments is mainly in the experiences gained and software modules developed to handle PHMR and XDS.b. One key aspect at present is augmenting measured values in the home with context information, for instance if a measurement is uploaded then who actually made the measurement: was it the patient alone, aided by a home nurse, or perhaps under the supervision of a trained clinician? This issue (which is irrelevant in a hospital setting) demonstrates how the symbiotic relationships between actors are important, as the issue was first raised by clinicians collaborating with Sekoia and thus brought to the attention of the architectural team for Net4Care and onwards to the larger group of actors. Thus the primary interest of the regional healthcare is learning and interaction.

Finally, the framework has been used by educational institutions, primarily for student master-level projects. The engineering school's projects had device integration as primary focus area and the framework thus supplied the back tier at a very modest learning cost. We have interviewed several of the groups and they agree that our QAS 1 (on page 25) is indeed fulfilled as getting the first "Hello World" application running is easy and the tutorials provide sufficient detail for getting their prototype work going without too much hassle. Also for their (simple) use, they found the framework's support for handling new types of measurements adequate, QAS 3. However, for adding context data as required by Sekoia, QAS 3 was not fully covered.

8. Discussion and future work

4S is now acting as an orchestrator and the Net4Care framework exhibits features of an ecosystem technical platform, especially with respect to the symbiotic relationships and interaction of the actors. However, we still await contributions in terms of software modules that can be used across the ecosystem. At present, software is built "on top" of Net4Care for the individual actor and largely for learning, testing, and experimentation, more than for contribution.

Additional studies of the applicability of the software ecosystem architecture concept to different ecosystems are required. The concept itself is not specific to telemedicine, and we would expect the types of analyzes and designs we have made to generalize to other types of ecosystems, i.e., that our study has external validity. We expect our work to be most useful in areas where there is no obvious orchestrator and/or no obvious candidate for doing the type of integrated analysis and design that the software ecosystem architecture approach supports. This view is supported by the fact that numerous sets of publicly funded open source software exists in Denmark, but their role and use is very limited, and no orchestrator exists. This is similar to the situation described in Section 5 and we find it likely that an analysis and subsequent design initiatives based on our approach will have a reasonable chance to improve the situation.

Reliability, i.e., the extent to which our study can be repeated is another probable issue. Since our case study is revelatory and our experiment depends on many parameters that are outside of our control (e.g., the involved actors and their behavior), it is questionable that it can be reproduced with the same results. However, as discussed above in relation to external validity, we find it likely that the software ecosystem architecture approach may be used successfully in ways similar to the work presented in this paper, but in different areas. This could be done e.g. in situations where funding is available for developing an open source platform, but no obvious orchestrator and/or no well-functioning organizational, business and/or software structure exist. In the following section we present a set of guidelines for people interested in this type of use.

8.1. Analysis and design guidelines

We provide a set of general guidelines on how a similar study can be carried out. We separate our guidelines in two sections: *analysis*, concerning the modeling of a software ecosystem, and *Design*, concerning the steps towards designing a software ecosystem.

8.1.1. Analysis

Phase A1: Identify and characterize the software ecosystem using the elements of the theory: technological platform, actors, and software build on the platform. Identify and characterize elements and relations of the three structures: organizational, business, and software. Select models to be used in describing them.

Questions to consider include:

- Are some expected types of actors, elements and/or relations missing, under- or overrepresented?
- Where in its evolution is the ecosystem (e.g., emerging, new, or old)?

Phase A2: Analyze and understand the software ecosystem: Collect and analyze information on the elements and relations of the three structures. Focus on the sustainability of each structure,

Table 3
List for actors in Net4Care.

Actor	Type	Size	Interest/contribution
Sekoia, Silverbullet	Companies	Small	Explore as integration system
Trifork/next step citizen	Company	Small	Use module; explore as integration system
CSC; Systematic	Companies	Large	Explore as integration system
CGI	Company	Large	XDS.b learning and integration
Region Nord; region Midt; region Hovedstaden	Hospitals	Large	PHMR and XDS.b learning
Aarhus University (School of Engineering); University of Southern Denmark	Universities	Large	Use as back-end
Aarhus University; University of Copenhagen	Universities	Large	Ecosystem/architectural case

and interactions among the structures, especially concerning issues.

Questions to consider include:

- Is the software ecosystem stable, unstable, or evolving?
- How are benefits distributed among the actors?
- What are the entry conditions and incentives to join?

Phase A3: Evaluate the results of analysis. Present and discuss results of analysis with actors and other stakeholders. Possibly repeat A1 and A2.

8.1.2. Design

Below we list the phases of designing an emergent ecosystem. The design is using the output of the analysis of the existing ecosystem. The steps are inspired by the steps for carrying out an experiment described by Wohlin et al. [63].

Phase D1: Scoping of design: Identify and characterize the goals of the design. Using the ecosystem analysis as input, focus on the areas of the ecosystem under development that appeared to be missing or problematic and define what purposes the design is going to serve. These purposes could relate to sustainability of the structures, e.g., overcoming issues identified through analysis.

Phase D2: Create a design. Develop a design consisting of one or more new or revised elements, e.g. software platform, orchestrator, keystone, or distribution channels. Develop a process plan for realizing the design. Since evolving an ecosystem is outside the control of any one actor, contingency planning should be considered.

Questions to consider include:

- Are the new/revised elements likely to create a process that will make the ecosystem meet the goals?
- How robust is the design, i.e. will all parts of the design have to realized?
- Who are the key actors whose participation is crucial for realization?

Phase D3: Evaluate and revise the design.

Discuss goals and design with key actors and other stakeholder, i.e., those whose support is crucial to the realization of the design.

Questions to consider include:

- Do actors agree with the goals?
- Do actors believe that realization of the design is feasible?
- Who will participate?
- Will the necessary resources be available?

Phase D4: Realize the design. Execute the plan and monitor the evolution of the ecosystem.

Phase D5: Evaluate the ecosystem. Model the software ecosystem after the design realization.

Questions to consider include:

- Have the identified areas improved?
- Has the general ecosystem health sustainability improved?
- Is the ecosystem providing more opportunities to the actors than before?
- Are there other areas that need improvement?

8.2. Future work

The recent addition of the Open Tele data collection system adds new groups of actors in addition to those listed in Table 3. The primary interest of the new groups of healthcare professionals and citizens/patients is to explore possibilities for improving and extending the system. Thus the first concrete activity will be a small series of evaluation and design workshops to provide input to further development of the Open Tele system.

Similarly, seminars are being planned for companies interested in the app store approach. In addition to software aspects of apps we have begun to look at healthcare and economic aspects. In one end of the spectrum is a category of apps where only traditional evaluations have been done, e.g., of stability and security issues. Somewhere in the middle are apps where clinical evaluations of effects have been carried out (and is made available through the app store). Finally we are considering a category of apps where citizens can be referred to receive them for free. These aspects are primarily related to the three administrative levels described in Section 5.1. 4S already has good relations to both the state and the regional levels and the relations to the level of municipalities are under development.

In addition to these new groups, several of the existing actors in Net4Care are also interested in Open Tele. Companies delivering systems to coordinate interaction among healthcare organizations providing service to patients having a chronic condition are considering Open Tele as a way to add data collection capabilities. However, for some of the companies making telemedicine systems Open Tele is a competitor. We expect that 4S will be able to provide a sufficient active and creative environment to make the pros more important than the cons. In any case we will have to be very explicit about the different roles of Net4Care and Open Tele in the ecosystem.

Finally, a number of municipalities are planning to acquire versions of Open Tele with a range of smaller additions/modifications. Currently the expectation is that the new/modified software will become part of the open source software managed by 4S.

9. Conclusion

The first contribution of this article has been to define the concept of 'software ecosystem architecture' as "the set of structures needed to reason about the software ecosystem, which comprise actor and software elements, relations among them, and properties of both"; identify the organizational, business, and software structures of software ecosystems as central; and apply these concept to the analysis of the current Danish telemedicine ecosystem, identifying challenges and opportunities in this.

The second contribution of the article has been to, within the framework of software ecosystem architecture, to present the design, realization, and (partial) evaluation of a software ecosystem, 4S, for telemedicine. The development of healthcare IT systems in general, and telemedicine applications in particular, is hindered by (i) complex problem and solution domains, by (ii) limited economies of scale, and by (iii) problems of integration and interoperability.

The 4S software ecosystem aims at (i) reducing the complexity of problem and solution domains by providing reusable components and services that incorporate national and international standards; at (ii) improving economies of scale by providing a platform (Net4Care and OpenTele) on which to build telemedical applications; and at (iii) increasing integration and interoperability by providing components to access national services in an interoperable way.

Finally, the Net4Care technological platform has showed how an emphasis on key success factors for application-centric software ecosystems, that of *simplified contribution* and *extensible data models and work flows*, can be supported by adopting a software engineering practice focusing on well-written on-line information resources, a staged testing environment, and an open source reference implementation, and thus serve as the software structure

basis for gaining interest, commitment, and symbiotic relationships among actors.

Acknowledgements

The research reported in this article has been supported by the Net4Care project that is funded by Caretech Innovation,[7] the Connect2Care project that is funded by the UNIK Partnership,[8] and the project "Kick-start of Denmark as telemedicine pioneer country" that is funded by The National Board of Technology and Innovation.[9]

References

[1] O. Barbosa, C. Alves, A systematic mapping study on software ecosystems, in: Third International Workshop on Software Ecosystems (IWSECO-2011), CEUR-WS, 2011, pp. 15–26.
[2] O. Barbosa, R.P. Santos, C. Alves, C. Werner, S. Jansen, Software ecosystems – analyzing and managing business networks in the software industry, in: Jansen et al. [29], Chapter: A Systematic Mapping Study on Software Ecosystems from a Three-Dimensional Perspective, 2013, pp. 59–81.
[3] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, second ed., Addison-Wesley, Boston, MA, USA, 2003.
[4] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, third ed., Addison-Wesley, Boston, MA, USA, 2013.
[5] K.W. Boone, The CDA Book, Springer, 2011.
[6] J. Bosch, From software product lines to software ecosystems, in: Proceedings of the 13th International Software Product Line Conference (SPLC '09), Carnegie Mellon University, Pittsburgh,PA, USA, 2009, pp. 111–119. <http://portal.acm.org/citation.cfm?id=1753235.1753251>.
[7] J. Bosch, Architecture challenges for software ecosystems, in: Proceedings of the Fourth European Conference on Software Architecture: Companion Volume (ECSA '10), ACM, New York, NY, USA, 2010, pp. 93–95. http://dx.doi.org/10.1145/1842752.1842776.
[8] J. Bosch, Architecture in the age of compositionality, in: M. Babar, I. Gorton (Eds.), Software Architecture, Lecture Notes in Computer Science, vol. 6285, Springer, Berlin/Heidelberg, 2010, pp. 1–4. doi:10.1007/978-3-642-15114-9_1.
[9] J. Bosch, P. Bosch-Sijtsema, Coordination between global agile teams: from process to architecture, in: Agility Across Time and Space, Springer, Berlin, Heidelberg, 2010, pp. 217–233. doi:10.1007/978-3-642-12442-6_15.
[10] J. Bosch, P. Bosch-Sijtsema, From integration to composition: on the impact of software product lines, global development and ecosystems, J. Syst. Softw. 83 (1) (2010) 67–76. SI: Top Scholars. <http://www.sciencedirect.com/science/article/B6V0N-4WPJ5XY-1/2/e69f658f21dfa50b9d1aa468a6cfb46d>.
[11] J. Bosch, P.M. Bosch-Sijtsema, Softwares product lines global development and ecosystems: collaboration in software engineering, in: Collaborative Software Engineering, Springer, Berlin, Heidelberg, 2010, pp. 77–92. doi:10.1007/978-3-642-10294-3_4.
[12] V. Boucharas, S. Jansen, S. Brinkkemper, Formalizing software ecosystem modeling, in: Proceedings of the 1st International Workshop on Open Component Ecosystems (IWOCE '09), ACM, New York, NY, USA, 2009, pp. 41–50. http://dx.doi.org/10.1145/1595800.1595807.
[13] C. Burkard, T. Widjaja, P. Buxmann, Software ecosystems, Business Inform. Syst. Eng. 4 (2012) 41–44. http://dx.doi.org/10.1007/s12599-011-0199-8.
[14] P.R.J. Campbell, F. Ahmed, A three-dimensional view of software ecosystems, in: Proceedings of the Fourth European Conference on Software Architecture: Companion Volume (ECSA '10), ACM, New York, NY, USA, 2010, pp. 81–84. http://dx.doi.org/10.1145/1842752.1842774.
[15] H.B. Christensen, M. Christensen, K.M. Hansen, M. Kyng, K. Manikas, M. Surrow, S. Urazimbetova, Requirements for a software-intensive ecosystem for telemedicine, in: Med@Tel 2012: Global Telemedicine and eHealth Updates, vol. 5, 2012, pp. 423–427.
[16] H.B. Christensen, K.M. Hansen, Net4care: towards a mission-critical software ecosystem, in: WICSA/ECSA, IEEE, 2012, pp. 224–228.
[17] J. Clemensen, S.B. Larsen, N. Ejskjaer, Telemedical treatment at home of diabetic foot ulcers, J. Telemed. Telecare 11 (suppl 2) (2005) 14–16.
[18] Danske Regioner, Regionernes fælles pejlemærker for digitalisering af sundhedsvæsenet. fra strategi til handling: Nye pejlemærker i perioden fra 2014 til 2016, 2013, in Danish.
[19] E. den Hartigh, M. Tol, W. Visscher, The health measurement of a business ecosystem, in: Proceedings of the European Network on Chaos and Complexity Research and Management Practice Meeting, 2006, pp. 1–39.
[20] N. Denzin, The Research Act: A Theoretical Introduction to Sociological Methods, McGraw-Hill, 1978.
[21] EgenJournal, CITH Co-constructing IT and Healthcare, 2010. <http://www.cith.dk>.
[22] FMK, Fælles medicinkort, 2010. <http://www.ssi.dk/Sundhedsdataogit/National%20Sundheds-it/Faelles%20Medicinkort.aspx>.
[23] Fonden for Velfærdsteknologi, National handlingsplan for udbredelse af telemedicin, June 2012, in Danish.
[24] K. Hansen, M. Ingstrup, M. Kyng, J. Olsen, Towards a software ecosystem of healthcare services, in: Proceedings of the 3rd International Workshop on Infrastructures for Healthcare: Global Healthcare, 2011. pp. 28–31.
[25] K.M. Hansen, The Net4Care platform – version 0.3, Tech. rep., Net4Care, 2012.
[26] HL7, HI7: health level seven international, 1987. <http://www.hl7.org>.
[27] M. Iansiti, R. Levien, The Keystone Advantage: What the New Dynamics of Business Ecosystems Mean for Strategy∗ Innovation∗ and Sustainability, Harvard Business Press, 2004.
[28] ISO 42010, Systems and software engineering – architecture description, ISO/IEC/IEEE 42010:2011(E), 2011.
[29] S. Jansen, S. Brinkkemper, M. Cusumano (Eds.), Software Ecosystems – Analyzing and Managing Business Networks in the Software Industry, Edward Elgar, Cheltenham, UK, 2013.
[30] S. Jansen, S. Brinkkemper, A. Finkelstein, Business network management as a survival strategy: a tale of two software ecosystems, in: First International Workshop on Software Ecosystems (IWSECO-2009), Citeseer, 2009, pp. 34–48.
[31] S. Jansen, S. Brinkkemper, A. Finkelstein, Software ecosystems – analyzing and managing business networks in the software industry, in: Jansen et al. [29], Chapter: Business Network Management as Survival Strategy, 2013, pp. 29–42.
[32] S. Jansen, S. Brinkkemper, J. Souer, L. Luinenburg, Shades of gray: opening up a software producing organization with the open software enterprise model, J. Syst. Softw. 85 (7) (2012) 1495–1510. <http://www.sciencedirect.com/science/article/pii/S0164121211003013>.
[33] S. Jansen, A. Finkelstein, S. Brinkkemper, A sense of community: a research agenda for software ecosystems, in: 31st International Conference on Software Engineering – Companion Volume, 2009 (ICSE-Companion 2009), May 2009, pp. 187–190.
[34] R. Kazman, M. Gagliardi, W. Wood, Scaling up software architecture analysis, J. Syst. Softw. 85 (7) (2012) 1511–1519. Software Ecosystems. <http://www.sciencedirect.com/science/article/pii/S0164121211000793>.
[35] T. Kilamo, I. Hammouda, T. Mikkonen, T. Aaltonen, From proprietary to open source-growing an open source ecosystem, J. Syst. Softw. 85 (7) (2012) 1467–1478. Software Ecosystems. <http://www.sciencedirect.com/science/article/pii/S0164121211001683>.
[36] T. Kilamo, I. Hammouda, T. Mikkonen, T. Aaltonen, Software ecosystems – analyzing and managing business networks in the software industry, in: Jansen et al. [29], Chapter: Open Source Ecosystem: A Tale of Two Cases, 2013, pp. 276–306.
[37] Kommunernes Landsforening, Kommunernes strategi for telesundhed, April 2013, in Danish.
[38] P.B. Kruchten, The 4 + 1 view model of architecture, IEEE Softw. 12 (6) (1995) 42–50.
[39] M. Lungu, M. Lanza, T. Gîrba, R. Robbes, The small project observatory: visualizing software ecosystems, Sci. Comput. Programm. 75 (4) (2010) 264–275. Experimental Software and Toolkits (EST 3): A special issue of the Workshop on Academic Software Development Tools and Techniques (WASDeTT 2008). <http://www.sciencedirect.com/science/article/B6V17-4X6MSPV-3/2200/91f82400b828c4fe2aa6bc04551d0a57>.
[40] K. Manikas, K.M. Hansen, Characterizing the Danish telemedicine ecosystem: making sense of actor relationships, in: Proceedings of the Fifth International Conference on Management of Emergent Digital EcoSystems, ACM, 2013, pp. 211–218.
[41] K. Manikas, K.M. Hansen, Reviewing the health of software ecosystems–a conceptual framework proposal, in: Fifth International Workshop on Software Ecosystems (IWSECO-2013), CEUR-WS, 2013, pp. 33–44.
[42] K. Manikas, K.M. Hansen, Software ecosystems – a systematic literature review, J. Syst. Softw. 86 (5) (2013) 1294–1306.
[43] J.D. McGregor, A method for analyzing software product line ecosystems, in: Proceedings of the Fourth European Conference on Software Architecture: Companion Volume (ECSA '10), ACM, New York, NY, USA, 2010, pp. 73–80. http://dx.doi.org/10.1145/1842752.1842773.
[44] MedCom, Overview of danish telemedical initiatives, 2013. <https://medcom.medware.dk/telemedicine_projects> (accessed December 2013).
[45] D. Messerschmitt, C. Szyperski, Software Ecosystem: Understanding an Indispensable Technology and Industry, MIT Press Books 1, 2003.
[46] National Sundheds-It, Referencearkitektur for opsamling af helbredsdata hos borgeren, Version 0.18 in public hearing, April 2013, in Danish.
[47] Net4Care, Net4care website, 2014. <http://www.net4care.org> (accessed April 2014).
[48] M. Olejaz, A.J. Nielsen, A. Rudkjøbing, H.O. Birk, A. Krasnik, C. Hernández-Quevedo, Denmark: health system review, Health Syst. Trans. 14 (2) (2012) 1–192.
[49] A. Osterwalder, The Business Model Ontology: A Proposition in a Design Science Approach, Ph.D. thesis, University of Lausanne, 2004.
[50] A. Osterwalder, Y. Pigneur, Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers, Wiley, 2010.
[51] M.Q. Patton, Qualitative Research and Evaluation Methods, SAGE Publications, Inc., 2002.
[52] PHMR, Implementation Guide for CDA Release 2.0 Personal Healthcare Monitoring Report (PHMR) (International Realm) Draft Standard for Trial Use Release 1.1, October 2010.

[7] http://www.caretechinnovation.dk.
[8] http://www.partnerskabetunik.dk.
[9] http://www.alexandra.dk/dk/projekter/sider/kickstart-af-danmark-som-telemedicinsk-foregangsland.aspx.

[53] R. Robbes, M. Lungu, A study of ripple effects in software ecosystems (nier track), in: Proceedings of the 33rd International Conference on Software Engineering (ICSE '11), ACM, New York, NY, USA, 2011, pp. 904–907. http://dx.doi.org/10.1145/1985793.1985940.
[54] RRS, Remote rehabilitation support, 2009. <http://www.caretechinnovation.dk/en/projects/rrs.htm>.
[55] R.P. Santos, C.M.L. Werner, A proposal for software ecosystem engineering, in: Third International Workshop on Software Ecosystems (IWSECO-2011), CEUR-WS, 2011, pp. 40–51.
[56] Sekoia, Sekoia website, 2014. <http://www.sekoia.dk> (accessed April 2014).
[57] TELEKAT, TELEKAT: Telehomecare, chronic patients and the integrated healthcare system, 2007. <http://www.telekat.eu>.
[58] Telesår, Telemedicinsk sårvurdering, 2006. <http://www.pleje.net>.
[59] S. Urazimbetova, Experiences of integration of telemedicine to national services, Tech. rep., Net4Care, 2012.
[60] J. van Angeren, V. Blijleven, S. Jansen, Relationship intimacy in software ecosystems: a survey of the dutch software industry, in: Proceedings of the International Conference on Management of Emergent Digital EcoSystems, MEDES '11, ACM, New York, NY, USA, 2011, pp. 68–75. http://dx.doi.org/10.1145/2077489.2077502.
[61] I. vanden Berk, S. Jansen, L. Luinenburg, Software ecosystems: a software ecosystem strategy assessment model, in: Proceedings of the Fourth European Conference on Software Architecture: Companion Volume, ECSA '10, ACM, New York, NY, USA, 2010, pp. 127–134. http://dx.doi.org/10.1145/1842752.1842781.
[62] ViewCare, Viewcare, 2011. <http://www.viewcare.com>.
[63] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering, Springer, 2012.
[64] B. Womack, Google says 700,000 applications available for android, October 2012. <http://www.businessweek.com/news/2012-10-29/google-says-700-000-applications-available-for-android-devices> (accessed April 2014).
[65] World Health Organization, Telemedicine. opportunities and developments in member states, Report on the second global survey on eHealth, 2010.
[66] XDS.b, IT Infrastructure Technical Framework, Volume 1 (ITI TF-1), Integration Profiles, Revision 9.0, August 2012.
[67] R.K. Yin, Case Study Research: Design and Methods, fifth ed., SAGE, 2013.